REMARKS

Applicant responds hereby to the office action dated September 28, 2007. Claims 3-5, 8-11, and 13-20 are amended hereby, claims 1-2, 6-7, 12, and 21-22 are cancelled without prejudice or disclaimer of subject matter. Claims 3-5, 8-11, and 13-20 remain pending hereinafter, where claims 3, 8, and 15 are independent claims.

Favorable consideration and allowance of the claims of the present application are respectfully requested.

Response to rejections under 35 U.S.C §103(a)

Claims 1-22 are rejected under 35 U.S.C §103(a) as allegedly being unpatentable over Ogg, et al. (US Patent No. 7,236,956) further in the view of Scheidt, et al. (US Patent No. 7,178,025).

In response, claims 1-2, 6-7, 12, 15, and 21-22 are cancelled without prejudice or disclaimer of subject matter. However, regarding claims 3-5, 8-11, and 13-20, applicants respectfully disagree in view of the amendments provided herein, and at least following:

the Examiner indicates in the Office Action that col.8, lines 10-29 in Ogg teaches, "the hash function comprises an exponentiation function". Respectfully, the SHA-1 function, as taught in Ogg, is a cryptographic hash function, but it does not comprise exponentiation function. Applicant provides the algorithm of SHA-1 that clearly shows SHA-1 does not include an exponentiation function.

```
Initialize variables:
h0 := 0x67452301
h1 := 0xEFCDAB89
h2 := 0x98BADCFE
h3 := 0x10325476
h4 := 0xC3D2E1F0

Pre-processing:
append the bit '1' to the message
```

```
append k bits '0', where k is the minimum number >= 0 such that the
resulting message
    length (in bits) is congruent to 448 (mod 512)
append length of message (before pre-processing), in bits, as 64-bit big-
endian integer

Process the message in successive 512-bit chunks:
break message into 512-bit chunks
for each chunk
    break chunk into sixteen 32-bit big-endian words w[i], 0 ≤ i ≤ 15

    Extend the sixteen 32-bit words into eighty 32-bit words:
    for i from 16 to 79
        w[i] := (w[i-3] xor w[i-8] xor w[i-14] xor w[i-16]) leftrotate 1

    Initialize hash value for this chunk:
    a := h0
    b := h1
    c := h2
    d := h3
    e := h4

    Main loop:
    for i from 0 to 79
        if 0 ≤ i ≤ 19 then
            f := (b and c) or ((not b) and d)
            k := 0x5A827999
        else if 20 ≤ i ≤ 39
            f := b xor c xor d
            k := 0x6ED9EBA1
        else if 40 ≤ i ≤ 59
            f := (b and c) or (b and d) or (c and d)
            k := 0x8F1BBCDC
        else if 60 ≤ i ≤ 79
            f := b xor c xor d
            k := 0xCA62C1D6

        temp := (a leftrotate 5) + f + e + k + w[i]
        e := d
        d := c
        c := b leftrotate 30
        b := a
        a := temp

    Add this chunk's hash to result so far:
    h0 := h0 + a
    h1 := h1 + b
    h2 := h2 + c
    h3 := h3 + d
    h4 := h4 + e

Produce the final hash value (big-endian):
digest = hash = h0 append h1 append h2 append h3 append h4
```

As seen in the SHA-1 algorithm, the main loop of the algorithm consists of logic. The algorithm only requires data manipulation such as exclusive or, or, and, not, addition, and rotating left. But, the algorithm does not execute any exponentiation function. The algorithm of SHA-1 can be found at http://en.wikipedia.org/wiki/SHA_hash_functions.

Though Ogg states at col. 8, lines 27-29 that RSA engines performs exponentiation function, RSA is an algorithm for public-key cryptography, not a hash function. Ogg also clearly states at col. 4, lines7-11 "**The encryption-decryption function** is employed using ... RSA public key encryption". Therefore, it is clearly understood that the RSA engines, as taught in Ogg, utilizes an RSA algorithm and performs an encryption-decryption function, not a hash function.

In response, claim 3 is amended to incorporate the subject matter of claims 1-2 and is now re-cast in an independent form. Claim 8 is amended to incorporate the subject matter of claims 6-7 and is now re-cast in an independent form. Claims 4-5, 11, and 13 are amended to depend on claim 3, which is patentably distinct as described above. Claims 9-10, and 14 are amended to depend on claim 8, which is patentably distinct as described above. Claim 15 is amended to add limitations from claims 1 and 3 as originally filed. Claims 16-20 depend on claim 15, which is now patentably distinct by the added limitation, "wherein the cryptographic function comprises a one-way hash function and the hash function comprises an exponentiation function". Therefore, in the virtue of dependency, dependent claims 4-5, 9-11, 13-14, and 16-20 are also patentably distinct.

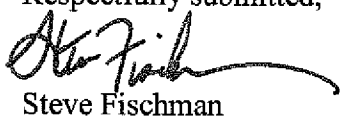
Respectfully, Scheidt does not disclose a hash function comprising an exponentiation function whether taken alone or in combination with Ogg. The Examiner is thus respectfully requested to withdraw the rejections of claims 1-22 under 35 U.S.C. §103(a).

Conclusion

Thus, the Examiner is respectfully requested to consider claims 3-5, 8-11, and 13-20 in light of the distinctions described in the above remarks, to allow these claims to proceed to issuance, which action is respectfully solicited.

In view of the foregoing, this application is now believed to be in condition for allowance, and a Notice of Allowance is respectfully requested.

Respectfully submitted,



Steve Fischman
Registration No. 34,594

Scully, Scott, Murphy & Presser, P.C.
400 Garden City Plaza, Suite 300
Garden City, New York 11530
(516) 742-4343

SF:JP:av